

Appendix A

```
END:      TCBWR  index
IDLE:     JMP  IDLE
LBLLISTEN:  SUB rcv[len], HEADERLEN --> seglen
            AND rcv[code], RST --> cond
            BRNEQZ IDLE
            AND rcv[code], ACK --> cond
            BRNEQZ TCPRST
            AND rcv[code], SYN --> cond
            BRNEQZ TCPINITWIN
            JMP TCPRST
LBLLISTEN1:  LOAD wkgreg[state] <-- SYNRCVD
            JMP END
LBLSYNRCVD:  JMP TCPSEQOK
LBLSYNRCVD1:  AND rcv[code], ACK --> cond
            BREQZ IDLE
            CMP wkgreg[suna], rcv[ack] --> cond
            BRNEQZ TCPRST
            CMP rcv[ack], wkgreg[snext] --> cond
            BRNEQZ TCPRST
            ADD wkgreg[suna], 1 --> wkgreg[suna]
            JMP TCPDATAPROC
LBLSYNRCVD4:  AND wkgreg[flags], RDONE --> R0
            EQUAL R0, 0 --> cond
            BRNEQZ LBLSYNRCVD5
            LOAD wkgreg[state] <-- CLOSEWAIT
            JMP END
LBLSYNRCVD5:  LOAD wkgreg[state] <-- ESTABLISHED
            JMP END
LBLESTABLISHED:  JMP TCPSEQOK
LBLESTABLISHED4:  CMP wkgreg[rbcoun], 75RBSIZE --> cond
            BREQZ LBLESTABLISHED5
            LOAD wkgreg[rbcoun] <-- 0
LBLESTABLISHED5:  AND wkgreg[flags], RDONE --> R0
            EQUAL R0, 0 --> cond
            BRNEQZ END
            LOAD wkgreg[state] <-- CLOSEWAIT
            JMP END
LBLCLOSEWAIT:  JMP TCPSEQOK
LBLLASTACK:  JMP TCPSEQOK
LBLLASTACK2:  AND rcv[code], ACK --> con
            BREQZ END
```

```

        CMP wkreg[suna], rcv[ack] --> cond
        BRNEQZ END
        CMP rcv[ack], wkreg[snext] --> cond
        BRNEQZ END
        JMP TCBDEALLOCATE
TCPRST:    LOAD snd>window] <-- 0
        AND rcv[code], SYN --> cond
        BREQZ LBL02
        ADD seglen, 1 --> seglen
LBL02:    AND rcv[code], FIN --> cond
        BREQZ LBL03
        ADD seglen, 1 --> seglen
LBL03:    AND rcv[code], ACK --> cond
        BRNEQZ LBL00
        LOAD snd[seq] <-- 0
        LOAD snd[code] <-- RST|ACK
        JMP LBL01
LBL00:    MOV rcv[ack] --> snd[seq]
        LOAD snd[code] <-- RST
LBL01:    ADD rcv[seq], seglen --> snd[ack]
        EQUAL wkreg[state], LISTEN --> cond
        BRNEQZ IDLE
        EQUAL wkreg[state], SYNRCVD --> cond
        BREQZ TCBDEALLOCATE
        AND rcv[code], SYN --> cond
        BRNEQZ TCBDEALLOCATE
        JMP IDLE
TCBDEALLOCATE:  CAM1CLR index
        JMP IDLE
TCPINITWIN:    LOAD wkreg[code] <-- SYN
        MOV rcv>window] --> wkreg[s>window]
        MOV rcv[seq] --> wkreg[lwseq]
        MOV rcv[seq] --> wkreg[rnext]
        ADD rcv[seq], RBSIZE --> wkreg[cwin]
        JMP TCPDATAPROC
TCPSENDWIN:    CMP wkreg[lwseq], rcv[seq] --> cond
        BRNEQZ LBL11
        EQUAL rcv[seq], wkreg[lwseq] --> cond
        BREQZ LBL10
        CMP wkreg[lwack], rcv[ack] --> cond
        BRNEQZ LBL11
LBL10:    MOV rcv>window] --> wkreg[s>window]
        MOV rcv[seq] --> wkreg[lwseq]
        MOV rcv[ack] --> wkreg[lwack]
LBL11:    EQUAL wkreg[state], ESTABLISHED --> cond

```

```

        BRNEQZ LBLESTABLISHED4
        OR wkgreg[flags], SNDFIN --> wkgreg[flags]
        JMP TCPACK
TCPSEQOK:    LOAD statusok <-- 0
            SUB rcv[len], HEADERLEN --> seglen
            AND rcv[code], SYN --> cond
            BREQZ LBL20
            ADD seglen, 1 --> seglen
LBL20:      AND rcv[code], FIN --> cond
            BREQZ LBL21
            ADD seglen, 1 --> seglen
LBL21:      SUB RBSIZE, wkgreg[rbcoun] --> rwindow
            EQUAL rwindow, 0 --> cond
            BREQZ LBL22
            EQUAL seglen, 0 --> cond
            BREQZ LBL22
            EQUAL wkgreg[rnext], rcv[seq] --> cond
            BREQZ LBL25
            LOAD statusok <-- 1
            JMP LBL25
LBL22:      EQUAL rwindow, 0 --> cond
            BRNEQZ LBL25
            ADD wkgreg[rnext], rwindow --> seqwin
            ADD rcv[seq], seglen --> seqlast
            EQUAL seglen, 0 --> cond
            BRNEQZ LBL23
            CMP seqlast, wkgreg[rnext] --> cond
            MOV cond --> statusok
            CMP seqlast, seqwin --> cond
            NOT cond --> cond
            AND cond, statusok --> statusok
LBL23:      CMP wkgreg[rnext], rcv[seq] --> cond
            BRNEQZ LBL25
            CMP seqwin, rcv[seq] --> cond
            OR statusok, cond --> statusok
LBL25:      AND rcv[code], SYN --> cond
            BREQZ LBL26
            SUB seglen, 1 --> seglen
LBL26:      AND rcv[code], FIN --> cond
            BREQZ LBL27
            SUB seglen, 1 --> seglen
LBL27:      EQUAL statusok, 0 --> cond
            BRNEQZ TCPACKOUT
            AND rcv[code], RST --> cond
            BRNEQZ TCBDEALLOCATE

```

```

    AND rcv[code], SYN --> cond
    BRNEQZ TCPRST
    EQUAL wkgreg[state], SYNRCVD --> cond
    BRNEQZ LBLSYNRCVD1
    JMP TCPACKIN
TCPACKOUT:    CMP seglen, 0 --> cond
    BRNEQZ LBL30
    AND rcv[code], SYN|FIN --> cond
    BREQZ IDLE
LBL30:        LOAD snd[code] <-- ACK
    MOV wkgreg[snext] --> snd[seq]
    MOV wkgreg[rnext] --> snd[ack]
    SUB RBSIZE, wkgreg[rbcoun] --> rwindow
    CMP wkgreg[state], SYNRCVD --> cond
    BREQZ LBL35
    SHL2 rwindow --> R0
    CMP RBSIZE, R0 --> cond
    BRNEQZ LBL32
    CMP RMSS, rwindow --> cond
    BREQZ LBL33
LBL32:        LOAD rwindow <-- 0
LBL33:        CMP wkgreg[cwin], wkgreg[rnext] --> cond
    BREQZ LBL34
    SUB wkgreg[cwin], wkgreg[rnext] --> R0
    CMP rwindow, R0 --> cond
    BRNEQZ LBL34
    MOV R0 --> rwindow
LBL34:        ADD wkgreg[rnext], rwindow --> wkgreg[cwin]
LBL35:        MOV rwindow --> snd[window]
    JMP END
TCPACK:        AND wkgreg[flags], SNDFIN --> R0
    EQUAL R0, 0 --> cond
    BRNEQZ LBL60
    OR wkgreg[code], FIN --> wkgreg[code]
LBL60:        OR wkgreg[code], ACK --> snd[code]
    AND wkgreg[flags], ~NEEDOUT --> wkgreg[flags]
    MOV wkgreg[snext] --> snd[seq]
    AND wkgreg[code], SYN|FIN --> cond
    BREQZ LBL61
    ADD wkgreg[snext], 1 --> wkgreg[snext]
LBL61:        MOV wkgreg[rnext] --> snd[ack]
    SUB RBSIZE, wkgreg[rbcoun] --> rwindow
    CMP wkgreg[state], SYNRCVD --> cond
    BREQZ LBL65
    SHL2 rwindow --> R0

```

```

    CMP RBSIZE, R0 --> cond
    BRNEQZ LBL62
    CMP RMSS, rwindow --> cond
    BREQZ LBL63
LBL62:    LOAD rwindow <-- 0
LBL63:    SUB wkgreg[cwin], wkgreg[rnext] --> R0
    CMP rwindow, R0 --> cond
    BRNEQZ LBL64
    MOV R0 --> rwindow
LBL64:    ADD wkgreg[rnext], rwindow --> wkgreg[cwin]
LBL65:    MOV rwindow --> snd>window]
    AND wkgreg[code], 0 --> wkgreg[code]
    EQUAL wkgreg[state], LISTEN --> cond
    BRNEQZ LBLLISTEN1
    EQUAL wkgreg[state], SYNRCVD --> cond
    BRNEQZ LBLSYNRCVD4
    EQUAL wkgreg[state], ESTABLISHED --> cond
    BRNEQZ TCPSENDWIN
    EQUAL wkgreg[state], CLOSEWAIT --> cond
    BREQZ END
    LOAD wkgreg[state] <-- LASTACK
    JMP END
TCPACKIN:    AND rcv[code], ACK --> cond
    BREQZ LBL41
    CMP rcv[ack], wkgreg[suna] --> cond
    BREQZ IDLE
    CMP rcv[ack], wkgreg[snext] --> cond
    BRNEQZ TCPACKOUT
    MOV rcv[ack] --> wkgreg[suna]
    AND wkgreg[code], SYN --> cond
    BREQZ LBL40
    AND wkgreg[code], ~SYN --> wkgreg[code]
    AND wkgreg[flags], ~FIRSTSEND --> wkgreg[flags]
LBL40:    AND wkgreg[code], FIN --> cond
    BREQZ LBL41
    EQUAL wkgreg[snext], rcv[ack] --> cond
    BREQZ LBL41
    AND wkgreg[code], ~FIN --> wkgreg[code]
    AND wkgreg[flags], ~SNDFIN --> wkgreg[flags]
LBL41:    EQUAL wkgreg[state], CLOSEWAIT --> cond
    BRNEQZ TCPSENDWIN
    EQUAL wkgreg[state], ESTABLISHED --> cond
    BRNEQZ TCPDATAPROC
    EQUAL wkgreg[state], LASTACK --> cond
    BRNEQZ LBLLASTACK2

```

```

        JMP END
TCPDATAPROC:  MOV rcv[code] --> statusok
               MOV rcv[seq] --> seqfirst
               AND statusok, URG --> cond
               BREQZ LBL51
               ADD seqfirst, rcv[urgptr] --> R0
               AND wkgreg[flags], RUPOK --> rwindow
               EQUAL rwindow, 0 --> cond
               BRNEQZ LBL50
               CMP R0, wkgreg[rupseq] --> cond
               BREQZ LBL51
LBL50:        MOV R0 --> wkgreg[rupseq]
               OR wkgreg[flags], RUPOK --> wkgreg[flags]
LBL51:        AND statusok, SYN --> cond
               BREQZ LBL52
               ADD wkgreg[rnext], 1 --> wkgreg[rnext]
               OR wkgreg[flags], NEEDOUT --> wkgreg[flags]
               ADD seqfirst, 1 --> seqfirst
LBL52:        SUB RBSIZE, wkgreg[rbcoun] --> rwindow
               ADD wkgreg[rnext], rwindow --> seqwin
               ADD seqfirst, seglen --> seqlast
               CMP wkgreg[rnext], seqfirst --> cond
               BREQZ LBL53
               SUB wkgreg[rnext], seqfirst --> R0
               SUB seglen, R0 --> seglen
               MOV wkgreg[rnext] --> seqfirst
LBL53:        CMP seqlast, seqwin --> cond
               BREQZ LBL54
               SUB seqlast, seqwin --> R0
               SUB seglen, R0 --> seglen
               AND statusok, ~FIN --> statusok
LBL54:        EQUAL seqfirst, wkgreg[rnext] --> cond
               BREQZ LBL55
               CMP seglen, 0 --> cond
               BREQZ LBL56
               ADD wkgreg[rnext], seglen --> wkgreg[rnext]
               ADD wkgreg[rbcoun], seglen --> wkgreg[rbcoun]
LBL512:       CAM2EMPTY cond
               BRNEQZ LBL511
               CAM2LLKUP seqlast
               BREQZ LBL511
               CAM2CLR [cam2_idx]
               ADD wkgreg[rnext], seglen --> wkgreg[rnext]
               ADD wkgreg[rbcoun], seglen --> wkgreg[rbcoun]
LBL511:       EQUAL wkgreg[finseq], wkgreg[rnext] --> cond

```

```

        BRNEQZ ALLDONE
        CMP wkgreg[pushseq], wkgreg[rnext] --> cond
        BRNEQZ NEXT
        OR statusok, PSH --> statusok
        LOAD wkgreg[pushseq] <-- 0
        JMP NEXT
ALLDONE:    OR statusok, FIN --> statusok
NEXT:      OR wkgreg[flags], NEEDOUT --> wkgreg[flags]
LBL56:     AND statusok, FIN --> cond
           BREQZ LBL58
           OR wkgreg[flags], RDONE|NEEDOUT --> wkgreg[flags]
           ADD wkgreg[rnext], 1 --> wkgreg[rnext]
LBL58:     AND statusok, PSH|URG --> cond
           BREQZ NEXTP1
           OR wkgreg[flags], PUSH --> wkgreg[flags]
           JMP NEXTP1
LBL55:     AND statusok, FIN --> cond
           BREQZ LBL59
           ADD seqfirst, seglen --> wkgreg[finseq]
LBL59:     AND statusok, PSH|URG --> cond
           BREQZ LBL510
           ADD seqfirst, seglen --> wkgreg[pushseq]
LBL510:    AND statusok, ~(FIN|PSH) --> statusok
           CAM2LLKUP seqlast
           BREQZ LBL515
           CAM2CLR [cam2_idx]
           ADD seqlast, seglen --> seqlast
           SUB seqlast, seqfirst --> seglen
LBL515:    CAM2RLKUP seqfirst
           BREQZ LBL516
           CAM2CLR [cam2_idx]
           SUB seqfirst, seglen --> seqfirst
           SUB seqlast, seqfirst --> seglen
LBL516:    CAM2WR seglen
           OR wkgreg[flags], NEEDOUT --> wkgreg[flags]
NEXTTP1:   AND wkgreg[flags], NEEDOUT --> R0
           EQUAL R0, 0 --> cond
           BREQZ TCPACK
           EQUAL wkgreg[state], LISTEN --> cond
           BRNEQZ LBLLISTEN1
           EQUAL wkgreg[state], SYNRCVD --> cond
           BRNEQZ LBLSYNRCVD4
           EQUAL wkgreg[state], ESTABLISHED --> cond
           BRNEQZ TCPSSENDWIN
           JMP END

```